Low Power and Area Consumption Custom Networks-On-Chip Architectures Using RST Algorithms

[1]P.Ezhumalai [2]Dr. Chilambuchelvan. A

[1] Dept of Computer Science Engineering
Ralalakshmi Engineering College, Thandalam-602 105, Chennai, India
[2]Department of Computer Science Engineering
R.M.K Engineering College, Chennai- 601 206
[1]ezhu.pubs@gmail.com , [2]chill97@gmail.com

Abstract*:* Network-on-Chip (NoC) architectures with optimized topologies have been shown to be superior to regular architectures (such as mesh) for application specific multiprocessor System-on-Chip (MPSoC) devices. The application specific NoC design problem takes, as input the system-level floorplan of the computation architecture .The objective is to generate an area and power optimized NoC topology. In this work, we consider the problem of synthesizing custom networks-on-chip (NoC) architectures that are optimized. Both the physical links and routers determine the power consumption of the NoC architecture. Our problem formulation is based on the decomposition of the problem into the inter-related steps of finding good flow partitions, and providing an optimized network implementation for the derived topologies. We used Rectilinear–Steiner-Tree (RST)-based algorithms for generating efficient and optimized network topologies. Experimental results on a variety of NoC benchmarks showed that our synthesis results were achieve reduction in power consumption and average hop count over different mesh implementations. We analyze the quality of the results and solution times of the proposed techniques by extensive experimentation with realistic benchmarks and comparisons with regular mesh-based NoC architectures.

Index Terms—Multicast routing, network-on-chip (NoC), synthesis, system-on-chip (SoC), topology**.**

1.Introduction

Network-on-Chip (NoC) is an emerging paradigm for communications within large VLSI systems implemented on a single silicon chip. The layered-stack approach to the design of the on-chip intercore communications is the Network-on-Chip (NOC) methodology. In a NoC system, modules such as processor cores, memories and specialized IP blocks exchange data using a network as a "public transportation" sub-system for the information traffic. A NoC is constructed from multiple point-to-point data links interconnected by switches (a.k.a. routers), such that messages can be relayed from any source module to any destination module over several links, by making routing decisions at the switches.

A NoC is similar to a modern telecommunications network, using digital bit-packet switching over multiplexed links. Although packet switching is sometimes claimed as necessity for a NoC, there are several NoC proposals utilizing circuit-switching techniques. This definition based on routers is usually interpreted so that a single shared bus, a single crossbar switch or a point-to-point network is not NoCs but practically all other topologies are. This is somewhat confusing since all above-mentioned are networks (they enable communication between two or more devices) but they are not considered as network-on-chips. Note that some erroneously use NoC as a synonym for mesh topology although NoC paradigm does not dictate the topology. Likewise, the regularity of topology is sometimes considered as a requirement, which is, obviously, not the case in research

concentrating on "application-specific NoC topology synthesis".

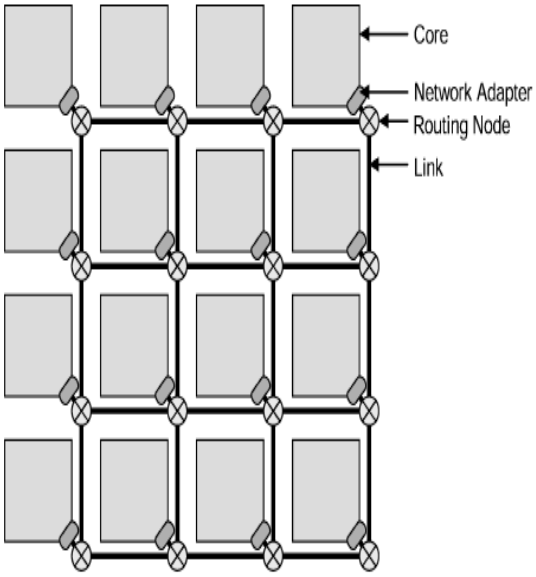


figure.1 Topological illustration of a 4-by-4 grid structured NoC.

The wires in the links of the NoC are shared by many signals. A high level of parallelism is achieved, because all links in the NoC can operate simultaneously on different data packets. Therefore, as the complexity of integrated systems keeps growing, a NoC provides enhanced performance (such as throughput) and scalability in comparison with previous communication architectures (e.g., dedicated point-to-point signal wires, shared buses, or segmented buses with bridges). Of course, the algorithms must be designed in such a way that they offer large parallelism and can hence utilize the potential of NoC.

Traditionally, ICs have been designed with dedicated point-to-point connections, with one wire dedicated to each signal. For large designs, in particular, this has several limitations from a physical design viewpoint. The wires occupy much of the area of the chip, and in nanometer CMOS technology, interconnects dominate both performance and dynamic power dissipation, as signal propagation in wires across the chip requires multiple clock cycles. NoC links can reduce the complexity of designing wires for predictable speed, power, noise, reliability, etc., because of their regular, well-controlled structure. From a system design viewpoint, with the advent of multi-core processor systems, a network is a natural architectural choice. A NoC can provide separation between computation and communication; support modularity and IP reuse via standard interfaces, handle synchronization issues, serve as a platform for system test, and, hence, increase engineering productivity.

Although NoCs can borrow concepts and techniques from the well-established domain of computer networking, it is impractical to blindly reuse features of "classical" computer networks and symmetric multiprocessors. In particular, NoC switches should be small, energy-efficient, and fast. Neglecting these aspects along with proper, quantitative comparison was typical for early NoC research but nowadays they are considered in more detail. The routing algorithms should be implemented by simple logic, and the number of data buffers should be minimal. Network topology and properties may be application-specific. Research on NoC is now expanding very rapidly, and there are several companies and universities that are involved. Figure 1 shows how a NoC, in comparison with shared buses, could be occupied with various components as resources

2.EXISTING RELATED WORKS

So far, the communication problems faced by System on chip were tackled by making use of regular Network on chip architectures. The following are the list of popular regular NoC architectures:

Mesh Architecture.
Torus Architecture.
Butterfly Fat Tree Architecture.
Extended Butterfly Fat Tree Architecture

The NoC design problem has received considerable attention in the literature. Towles and Dally [1] and Benini and De Micheli [2] motivated the NoC paradigm. Several existing NoC solutions have addressed the mapping problem to a regular mesh-based NoC architecture [3], [4]. Hu and Marculescu [3] proposed a branch-and-bound algorithm for the mapping of computation cores on to mesh-based NoC architectures. Murali *et al.* [4] described a fast algorithm for mesh-based NoC architectures that considers different routing functions, delay constraints, and bandwidth requirements. On the problem of designing custom NoC architectures without assuming existing network architecture, a number of techniques have been proposed [5]–[10]. Pinto *et al.* [7] presented techniques for the constraint-driven communication architecture synthesis of point-to-point links by using heuristic-based -way merging. Their technique is limited to topologies with specific structures that have only two routers between each source and sink pair. Ogras *et al.* [5], [6] proposed graph decomposition and long link insertion techniques for application-specific NoC architectures. Srinivasan *et al.* [8], [9] presented NoC synthesis algorithms that consider system-level floor planning, but their solutions only considered solutions based on a slicing floorplan where router locations are restricted to corners of cores and links run around cores. Murali *et al.* [10] presented an innovative deadlock-free NoC synthesis flow with detailed backend integration that also considers the floorplanning process. The proposed approach is based on the min-cut partitioning of cores to routers. This work presents a synthesis approach based on a set partitioning formulation that considers multicast traffic, Although different in topology and some other aspects, all the above papers essentially advocate the advantages of using NoCs and regularity as effective means to design high performance SoCs. While these papers mostly focus on the concept of regular NoC architecture (discussing the overall advantages and challenges), to the best of our knowledge, our work is better than previous custom NoC synthesis formulations and efficient way to solve it.

PROPOSED SYSTEM

3.1 PROBLEM DEFINITION

- We consider the problem of synthesizing custom networks-on-chip (NoC) architectures that are optimized for a given application.
- We divide the problem statement into the flowing interrelated steps:

  Physical topology Construction.
  Power and Area Comparisons

3.2 SYSTEM ARCHITECTURE

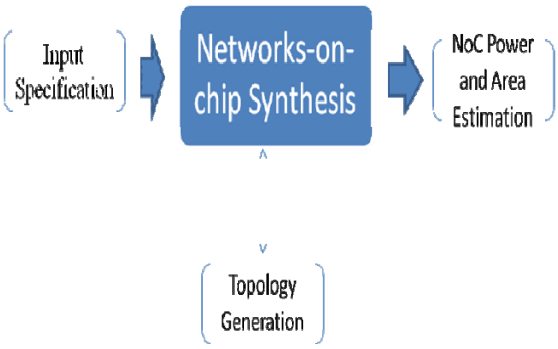


Figure. 2 Proposed System Architecture

Our NoC synthesis design flow is depicted in Figure 2. The major elements in the design flow are elaborated as follows.

Input Specification*:* The input specification to our design flow consists of a list of modules. As observed in recent trends, many

modern SoC designs combine both hard and soft modules as well as both packet-based network communications and conventional wiring. Modules can correspond to a variety of different types of intellectual property (IP) cores such as embedded microprocessors, large embedded memories, digital signal processors, graphics and multimedia processors, and security encryption engines, as well as custom hardware modules. These modules can come in a variety of sizes and can be either hard or soft macros, possibly as just black boxes with area and power estimates and constraints on aspect ratios. To facilitate modularity and interoperability of IP cores, packet-based communication with standard network interfaces is rapidly gaining adoption. Custom NoC architectures are being advocated as a scalable solution to packet-based communication. In general, a mixture of network-based communications and conventional wiring may be utilized as appropriate, and not all inter-module communications are necessarily over the on-chip network. For example, an embedded microprocessor may have dedicated connections to its instruction and data cache modules. Our design flow and input specification allow for both interconnection models. Below is an example of a communication demand graph:

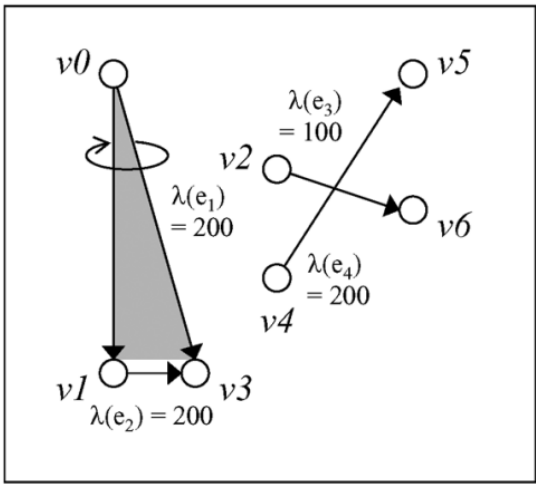
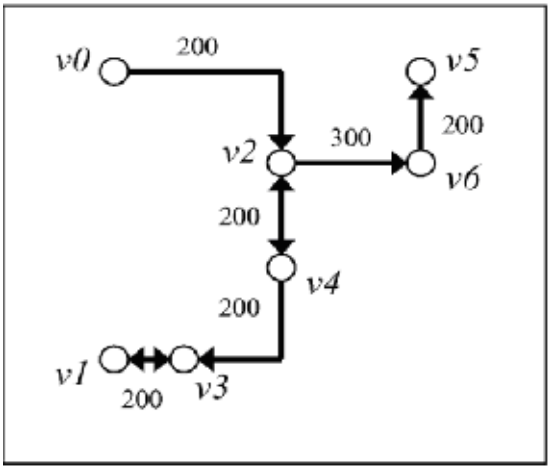


Figure 3 Sample Input Specification

NoC Synthesis*:* Given input specification information, the NoC synthesis step then proceeds to synthesize a NoC architecture that is optimized for the given specification. Consider the above diagram that depicts a small illustrative example. It only shows the portion of the input specification that corresponds to the network-attached modules and their traffic flows. The nodes represent modules, edges represent traffic flows, and edge labels represent the length of the two vertices. The NoC Synthesis generates topologies based on the communication demand graph and comparing with parameters like power consumption and area usage chooses the best architecture. Below is an example of two architectures generated based on the given CDG.



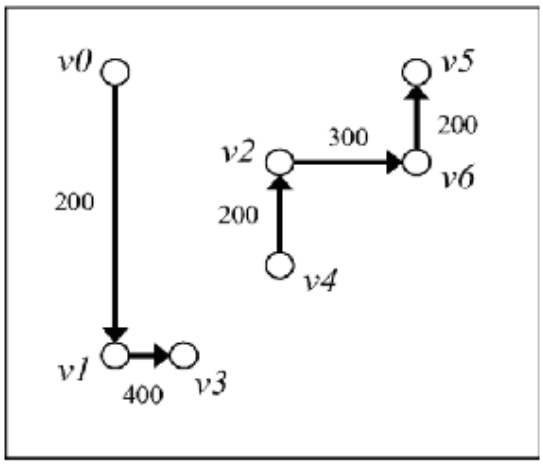


Figure 4 Sample Topologies Generated

NoC Power and Area Estimation*:* To evaluate the power and area of the synthesized NoC architecture, we use a state-of the- art NoC power-performance simulator called Orion that can provide detailed power characteristics for different power components of a router for different input/output port configurations. It accurately considers leakage power as well as dynamic switching power, which is important since it is well known that leakage power is becoming an increasingly dominating. Orion also provides area estimates based on a state-of-the-artrouter microarchitecture.

MODULE DESCRIPTION

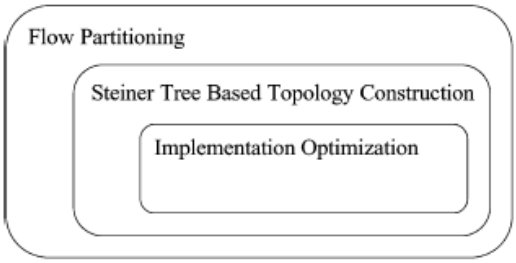


Figure 5 Formulation of Synthesis Problem

4.1 Flow Partitioning

Flow partitioning is performed in the outer loop of our synthesis formulation to explore different partitioning of flows to separate subnetworks. We make use of the following algorithm to implement flow partitioning:

4.2 STEINER TREE BASED TOPOLOGY CONSTRUCTION

For each flow partition considered, physical network topologies must be decided. In current process technologies, layout rules for implementing wires dictate physical topologies where the network links run horizontally or vertically. Thus, the problem is similar to Rectilinear Steiner Tree (RST) problem that has been extensively studied for the conventional VLSI routing problem. Given a set of nodes, the RST problem is to find a network with the shortest edge lengths using horizontal and vertical edges such that all nodes are interconnected. The RST problem is well studied with very fast implementations available. We create an RST solver in the inner loop of flow partitioning to generate topologies for the set partitions considered.

Input: $G(V, E, \pi, \lambda)$: communication demand graph
C: specified evaluation function for implementation cost
Output: T: synthesized network architecture
initialize $P^0 = \emptyset$
for all $e_k \in E$
  $P^0 = P^0 \cup \{e_k\}$
  $\text{cost}(\{e_k\}) = \text{EvaluateCost}(T(\{e_k\}), C)$
end for
$t - 0$
while $P^t| > 1$ **do**
  for all $g_u, g_v \in P^t$ **do**
    $g_{uv} = g_u \cup g_v$
    $T(g_{uv}) = \text{SolveRST}(g_{uv})$
    $\text{cost}(g_{uv}) = \text{EvaluateCost}(T(g_{uv}), C)$
    $\beta(g_u, g_v) = \text{cost}(g_{uv}) + \sum_{g_i \in P^t, g_i \neq g_u, g_v} \text{cost}(g_i)$
  end for
  $(u, v) = \arg\min_{g_u, g_v \in P^t} \beta(g_u, g_v)$
  $P^{t+1} = P^t \setminus \{g_u, g_v\}$
  $P^{t+1} = P^{t+1} \cup \{g_u \cup g_v\}$
  $t = t + 1$
end while
for all $t \in [0, n-1]$ **do**
  $c(P^t) = \sum_{g_u \in P^t} \text{cost}(g_u)$
  $soln[P^t] = \bigcup_{g_u \in P^t} T(g_u)$
end for
$t - \arg\min_{t \in [0,n-1]} c(P^t)$
$T - soln[P^t]$
return T

Figure 6 Flow Partitioning Algorithm

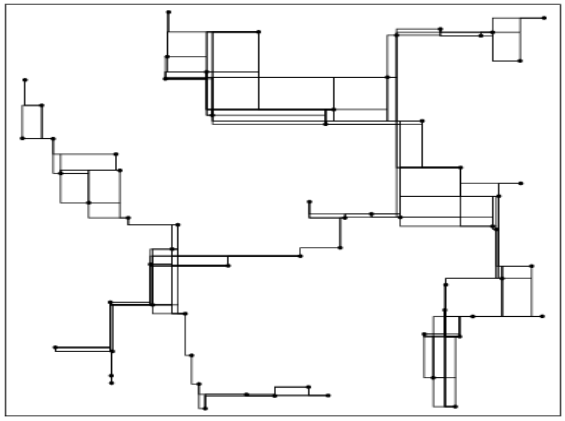
IMPLEMENTATION RESULTS

5.1. EXPERIMENTAL SETUP

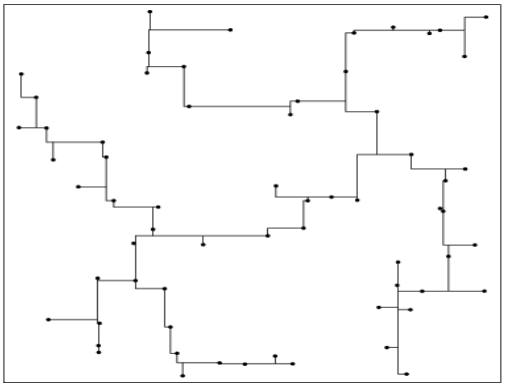
We have implemented our proposed algorithm in C. In our implementation, we have designed a Rectilinear Steiner Tree solver to generate the physical network topologies in the inner loop of the algorithm. Simulator ORION 2.0 does the power and area estimates. The Results obtained are shown in a line chart for mere comparisons. A snapshot of the all the results have been shown later in this chapter. All experimental results were obtained on a

3.06-GHz Intel P4 processor machine with 512 MB of memory running Linux.

5.2. EXPERIMENTAL RESULTS



ALL FSTs: 64 Points
Figure 7 Snapshot of ALL The FSTs Generated



Steiner Minimal Tree: 64 Points, length = 56729

Figure 8 Steiner Minimal Tree Generated

Method of Evaluation: In all our experiments, we aim to evaluate the performance of the proposed algorithms. On all benchmarks with the objective of minimizing the total area as well as power consumption of the synthesized NoC architectures. The total area as well as power consumption includes all network components. We applied the design parameters of 1 GHz clock frequency, 4-flit buffers, and 128-bit flits. For evaluation, fair direct comparison with previously published NoC synthesis results is difficult in part because of vast differences in the parameters assumed. To evaluate the effectiveness of our algorithms, we have the full mesh implementation for each benchmark for comparison from previous published papers have been taken. These comparisons are signified to show the benefits of custom NoC architectures.

Table 1. NOC Power Comparisons

S.No	Vertices	Custom Power	Mesh Power	Opt. Mesh Power
1	6	0.0416	0.0990	0.0430
2	7	0.0432	0.1000	0.0500
3	8	0.0494	0.1780	0.0600
4	11	0.0617	0.2570	0.1220
5	12	0.0663	0.2720	0.1520
6	14	0.0848	0.3760	0.1310
7	20	0.0987	0.4950	0.1540
8	24	0.1034	0.6330	0.2020
9	25	0.1034	0.6420	0.2600
10	36	0.1203	1.0080	0.3000
11	44	0.1358	1.4250	0.3440

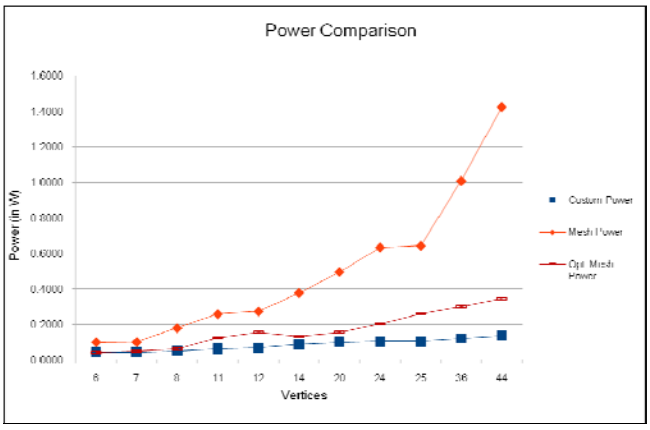


Figure 9 NoC Power Comparisons

The area results, power results, the execution times, and area as well as power improvements of that algorithm are reported. The results show the algorithm can efficiently synthesize NoC architectures that minimize power and area consumption as

compared with regular topologies such as mesh and optimized mesh topologies.

Table 2. NoC Area Comparisons

S.No	Vertices	Custom Area	Opt. Mesh Area
1	6	0.1543	0.31
2	7	0.1557	0.43
3	8	0.1810	0.41
4	12	0.2252	0.48
5	14	0.3060	0.91
6	20	0.3563	1.03
7	24	0.3732	1.19
8	25	0.3753	1.81
9	36	0.4382	1.81
10	44	0.4936	2.01

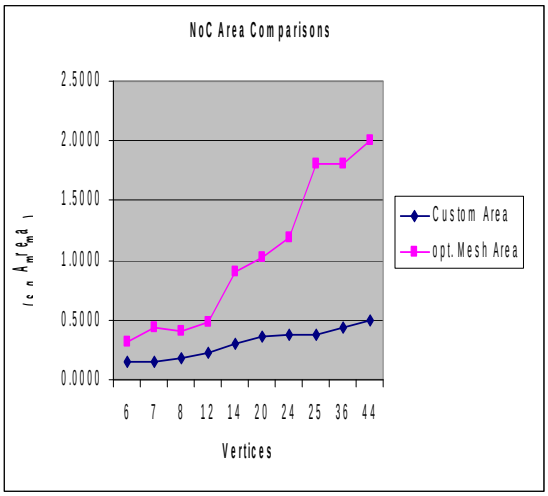


Figure 10. NoC Area Estimates

Thus, the above two line charts in figure 9 and 10 clearly show a reduction in power and area estimates of custom NoC with mesh and optimized mesh topologies. Mesh topologies was explained in chapter 2. Eliminating router ports and links that are not used forms optimized mesh topologies. The power reduction is at an average of 83.43 percent and 50 percent as compared to mesh and optimized mesh topologies respectively. The area reduction is at an average of 70.95 percent as compared to optimized mesh topologies.

6.CONCLUSION AND FUTURE WORK

In this research Works have been carried out in context related to Regular topologies like mesh, torus and etc. This work presented an idea on building customizing network on chip with the better flow partitioning and also considered power and area reduction as compared to the already presented Regular topologies, we proposed a formulation of the custom NoC synthesis problem based on the decomposition of the problem into the inter-related steps of deriving a good physical network topology, and providing an comparison in terms of area and power with the well established regular topologies. We used the algorithm called CLUSTER for systematically examining different possible set partitioning of flows, and we proposed the use of RST algorithms for constructing good physical network topologies. Our solution framework enables the decoupling of the evaluation cost function from the exploration process, thereby enabling different user objectives and constraints to be considered. Although we use Steiner trees to generate a physical network topology for each group in the set partition, the final NoC architecture synthesized is not necessarily limited to just trees as Steiner tree implementations of different groups may be connected to each other to form non-tree structures.

This work does not differentiate the routers/switches (communication modules) with the operating modules present in the chip. In near future, the work of identifying the best placement of routers and minimizing the number of routers and also the effectives of the customized Network on Chip in terms of other parameters like throughput, latency.

Link utilization and buffer utilization can be taken into account.

REFERENCES

[1] Shan Yan, Bill Lin, " Custom Networks-on-Chip Architectures With Multicast Routing," IEEE transactions on very large scale integration (VLSI) systems, vol. 17, no. 3, march 2009.
.

[2] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear-programming based techniques for synthesis of network-on-chip architectures," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 14, no. 4, pp. 407–420, Apr. 2006.

[3] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Application specific network-on-chip design with guaranteed quality approximation algorithms," in Proc. ASPDAC, 2007, pp. 184–190.

[4] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, G . De Micheli, and L. Raffo, "Designing application-specific networks on chips with floor plan information," in Proc. ICCAD, 2006, pp. 355–362.

[5] L. Zhang, H. Chen, H. Chen, B. Yao, K. Hamilton, and C.-K. Cheng, "Repeated on-chip interconnect analysis and evaluation of delay, power, and bandwidth metrics under different design goals," in Proc. ISQED, 2007, pp. 251–256.

[6] R. Mullins, "Minimizing dynamic power consumption in on-chip networks," in Proc. Int. Symp. Syst.-on-Chip, 2006, pp. 1–4.

[7] C. -W. Lin, S. -Y. Chen, C. -F. Li, Y. -W. Chang, and C. -L. Yang, "Efficient obstacle-avoiding rectilinear Steiner tree construction," in Proc.Int. Symp. Phys. Des. 2007, pp. 127–134.

[8] D. Greenfield, A. Banerjee, J. -G. Lee, and S. Moore, "Implications of rent's rule for NoC design and its fault-tolerance," in Proc. NOCS, May 2007, pp. 283–294.

[9] S. Yan and B. Lin, "Application-specific network-on-chip architecture synthesis based on set partitions and Steiner trees," in Proc. ASPDAC, 2008, pp. 277–282.

[10] Xilinx, San Jose, CA, "UMC delivers leading-edge 65 nm FPGAs toXilinx," Des. Reuse, Nov. 8, 2006 [Online]. Available: http://www.design-reuse.com/news/14644/umc-edge-65nm-fpgas-xilinx.html

[11] P. Gratz, K. Sankaralingam, H. Hanson, P. Shivakumar, R.McDonald, S. W. Keckler, and D. Burger, "Implementation and evaluation of a dynamically routed processor operand network," in Proc. NOCS, May 2007, pp. 7–17.

[12] N. Enright-Jerger, M. Lipasti, and L.-S. Peh, "Circuit-switched coherence," IEEE Computer. Arch. Lett. vol. 6, no. 1, pp. 193–202, Mar. 2007.

[13]. Shan Yan, Student Member, IEEE, and Bill Lin, Senior Member, IEEE "Custom Networks-on-Chip Architectures With Multicast Routing" IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 17, No. 3, Pp 342-355, March 2009.

Ezhumalai Periyathambi received the B.E degree in Computer Science and engineering from Madras University, Chennai , India in 1992 and Master Technology

(M.Tech.,) in computer science and Engineering from J N T University, Hyderabad, India in 2006. He is currently working towards the Ph.D degree in Department of Information and Communication, Anna University, Chennai, India. He is working as Professor in the Department of Computer Science and Engineering , Rajalakshmi Engineering College, Chennai, Tamilnadu, India. His research in reconfigurable architecture, Multi-Core Technology CAD – Algorithms for VLSI Architecture. Theoretical Computer Science. And mobile computing.